Ollscoil na Gaillimhe
University of Galway

CT336/CT404
Graphics & Image Processing

Sem 1 (Autumn) 2024-25,
Dr. Nazre Batool

4th year B.Sc. (CS&I.T.).
2nd year M.Sc. (Software Development & Design)
1st year M.Sc. (Biomedical Engineering)
Visiting students

University
*of*Galway.ie

Lecture 8: Segmentation & Model Fitting

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- Last time – started understanding more complex operations on images:
    - Multiresolution processing & Image Pyramids
    - Edges
    - Feature Detectors & Feature Descriptors
    - Segmentation (Part 1 of 2)

- Today:
  - Image Segmentation (Part 2 of 2)
    - Thresholding (Global and Adaptive)
    - Overview of other techniques

  - Model Fitting/Template Matching
    - Hough (pronounced 'Huff') Transform

- **Acknowledgement:** These lecture slides have been prepared using digital materials available with your Text book: 'Image Processing and Analysis by Stan Birchfield, 1st Edition, Cengage Learning'

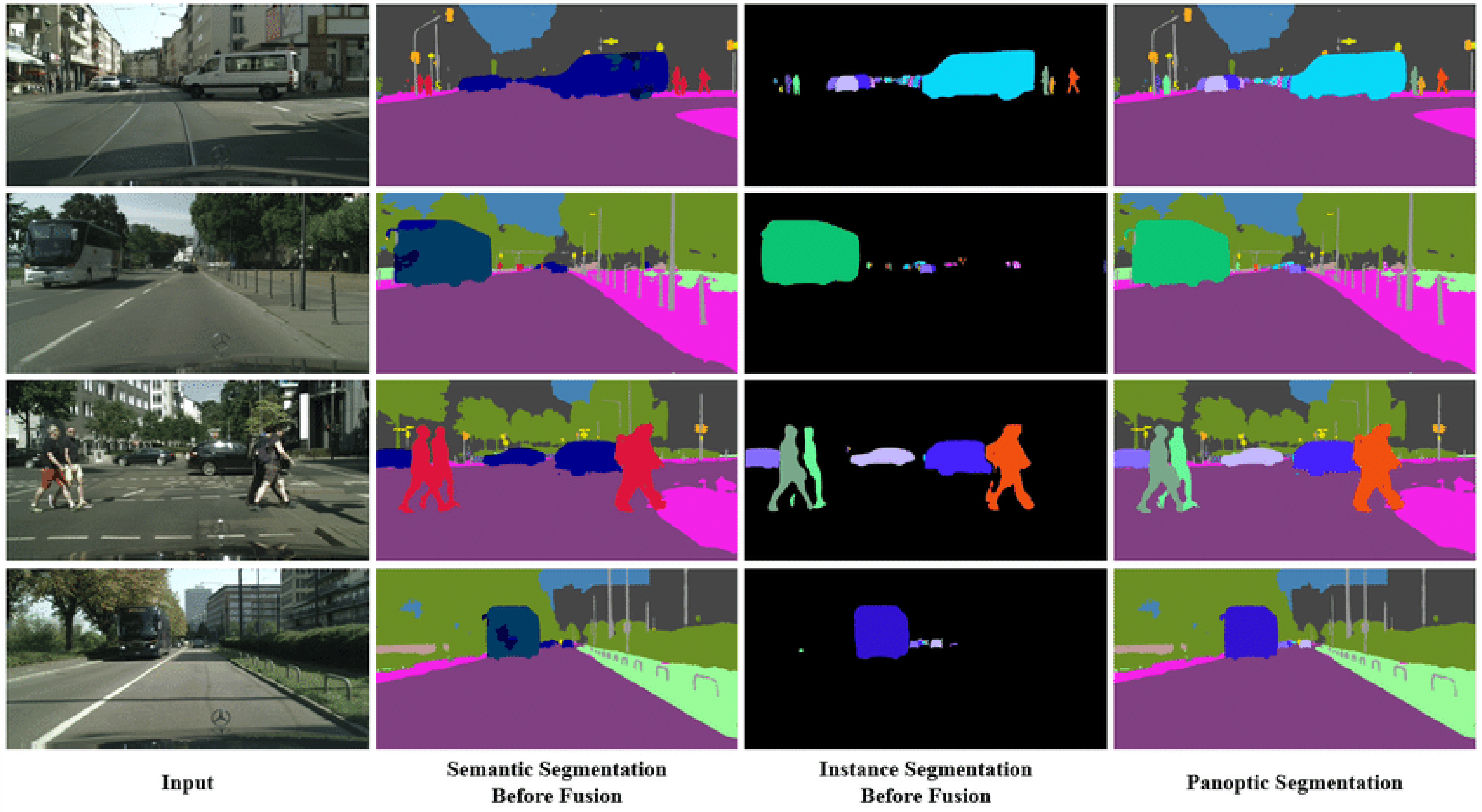
Image Segmentation

- A fundamental image **analysis** task
- A ~~bottom-up~~ process that groups pixels in an image based on their low-level properties such as colour or texture.

- Three related tasks defined in recent years:

  - **Semantic Segmentation:** labels every single pixel contained in an image by its semantic class
  - **Instance Segmentation:** focuses only on the semantic classes that can be counted
  - **Panoptic Segmentation:** entails both of the above

  - *Have you encountered any segmentation technique so far?*



Fusion Scheme for Semantic and Instance-level Segmentation – Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Panoptic-segmentation-by-unifying-semantic-and-instance-segmentation_fig2_329616112 [accessed 23 Oct 2024]

Image Segmentation

- **Thresholding:** To segment foreground object(s) from background, binary classification
- **Global Thresholding:** Same threshold value, **once determined,** is used for the complete image. Does NOT emphasise neighbourhood properties.
- Two simple, widely used global thresholding technique are known as the **Ridler-Calvard** algorithm and **Otsu's method**.

$$I'(x,y) = \begin{cases} \text{ON} & \text{if } I(x,y) > T \\ \text{OFF} & \text{otherwise} \end{cases}$$

Global Thresholding

Figure 10.1 LEFT: A grayscale image of several types of objects (fruit) on a dark background (conveyor belt). RIGHT: The graylevel histogram of the image.

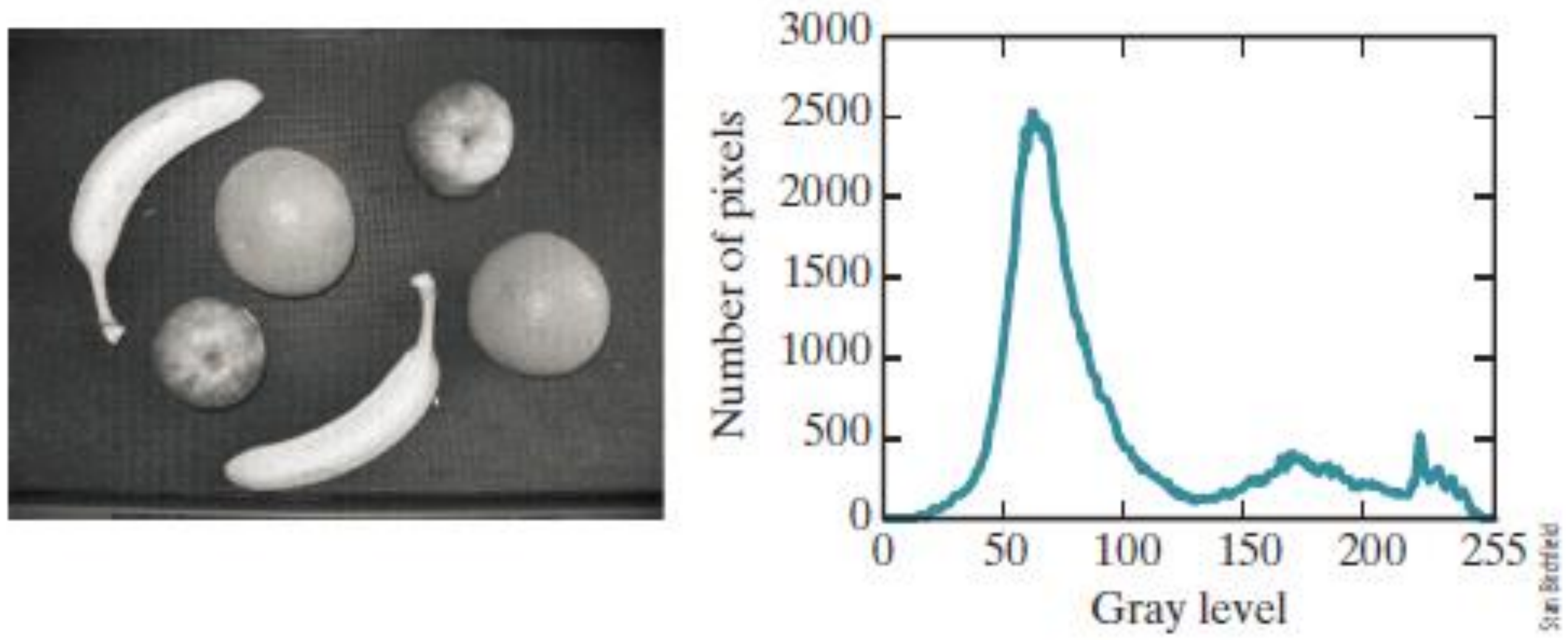


Image Segmentation

- **Global Thresholding Ridler-Calvard** Algorithm for binary classification**:**

- Let τ be a threshold, and let μ_⊲ be the mean gray level of all the pixels whose gray level is less than or equal to τ, while μ_⊳ is the mean gray level of all the pixels whose gray level is greater than τ. The algorithm determines the optimal mean values in terms of 'image moments' m_0 and m_1.

$$\mu_{\blacktriangleleft} = \frac{m_1[\tau]}{m_0[\tau]} \quad \text{and} \quad \mu_{\triangleright} = \frac{m_1[\zeta - 1] - m_1[\tau]}{m_0[\zeta - 1] - m_0[\tau]} \tag{10.2}$$

$$m_0[\tau] = \sum_{\ell=0}^{\tau} h[\ell] \quad \text{and} \quad m_1[\tau] = \sum_{\ell=0}^{\tau} \ell h[\ell] \tag{10.3}$$

- Where image moments are defined in terms of gray level histograms

Image Segmentation

- **Global Thresholding Ridler-Calvard** Algorithm:
- Iterate between step 1 and 2 (similar to the ‘Expectation-Maximization algorithm’)
  - Step 1 (lines 9, 10): Computes the two means based on the current estimate of threshold.
  - Step 2 (line 11): Set the threshold to the average of the two new means.
- Assumes that foreground and background gray levels are distributed as Gaussians with equivalent variance (and standard deviation).

ALGORITHM 10.1 Compute an image threshold using the Ridler-Calvard algorithm

RIDLER-CALVARD(I)

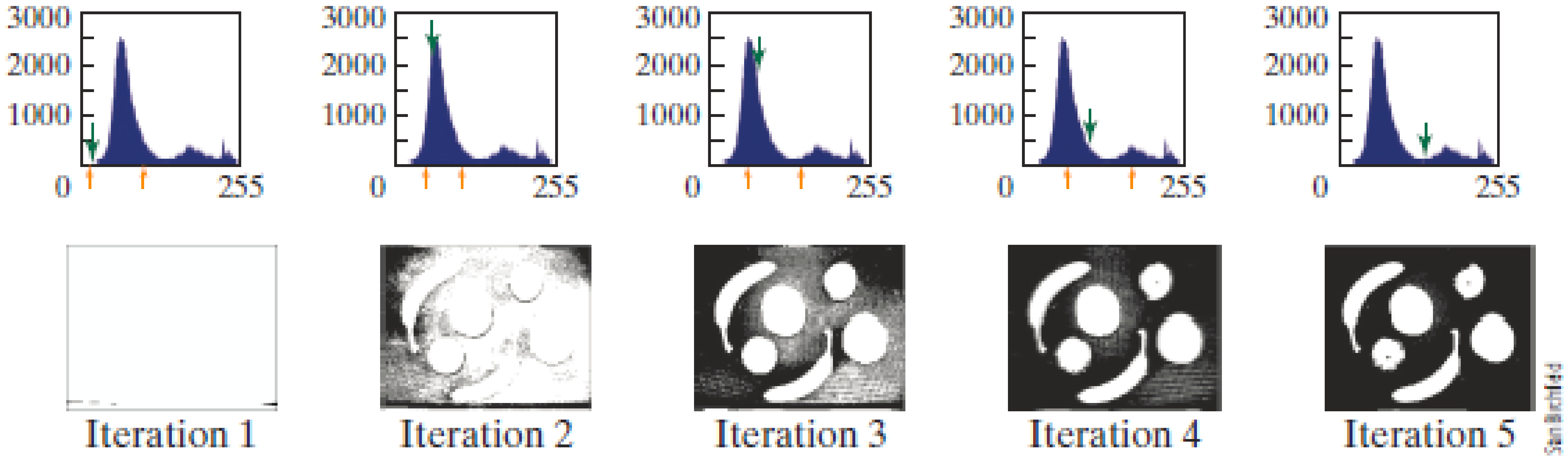
Input: grayscale image I
Output: threshold value τ

1 $h \leftarrow$ COMPUTEHISTOGRAM(I)
2 $m_0[0] \leftarrow h[0]$
3 $m_1[0] \leftarrow 0$
4 **for** $k \leftarrow 1$ **to** $\zeta - 1$ **do**
5 $\quad m_0[k] \leftarrow m_0[k-1] + h[k]$
6 $\quad m_1[k] \leftarrow m_1[k-1] + k * h[k]$
7 $\tau \leftarrow \zeta/2$
8 **repeat**
9 $\quad \mu_{\blacktriangleleft} \leftarrow m_1[\tau]/m_0[\tau]$
10 $\quad \mu_{\triangleright} \leftarrow (m_1[\zeta-1] - m_1[\tau])/(m_0[\zeta-1] - m_0[\tau])$
11 $\quad \tau \leftarrow$ ROUND $\left(\frac{1}{2}(\mu_{\blacktriangleleft} + \mu_{\triangleright})\right)$
12 **until** τ does not change
13 **return** τ

Image Segmentation

- **Example: Global Thresholding Ridler-Calvard** Algorithm:

Figure 10.2 Step-by-step example of the Ridler-Calvard algorithm applied to the image of Figure 10.1. Note that even with an initial threshold far from the true solution, the algorithm converges in only five iterations. The top row shows the histogram. The green arrow pointing down indicates the threshold at each iteration, while the gold arrows pointing up indicate the two means. The bottom row shows the result of thresholding the image using the threshold for that iteration.



- *What if the variances are different? Otsu's Method*

Image Segmentation

- **Global Thresholding: Otsu's Method**

Assuming that the underlying distributions can have different variances, find a threshold that minimizies the 'within-class' variance which is defined as the weighted sum of the variances of the two groups of pixels:

$$\sigma_w^2(\tau) \equiv p_{\blacktriangleleft}(\tau)\sigma_{\blacktriangleleft}^2(\tau) + p_{\triangleright}(\tau)\sigma_{\triangleright}^2(\tau) \qquad (10.6)$$

- Compare this with 'between-class' variance:

$$\sigma_b^2(\tau) \equiv p_{\blacktriangleleft}(\tau)(\mu_{\blacktriangleleft}(\tau) - \mu)^2 + p_{\triangleright}(\tau)(\mu_{\triangleright}(\tau) - \mu)^2 \qquad (10.10)$$

- Total variance = 'within-class' variance + 'between-class' variance ($\sigma^2 = \sigma_w^2 + \sigma_b^2$)

- Since the total variance σ^2 does not depend on the threshold, minimizing σ_w^2 is the same as maximizing σ_b^2. The advantage of the latter is that it is dependent only upon first-order properties (means) rather than second-order properties (variances), thus making it easier to compute

$$\sigma_b^2(\tau) = \frac{(m_1[\tau] - \mu m_0[\tau])^2}{m_0[\tau](m_0[\zeta - 1] - m_0[\tau])} \qquad (10.16)$$

Image Segmentation

▪ Global Thresholding Otsu's Method

Since there is a small number of possible thresholds (usually just 256), Otsu's method
iterates through all these possible values for to find the one that maximizes the quantity σ_b^2.

ALGORITHM 10.2 Compute an image threshold using Otsu's method

```
OTSU(I)

Input:   grayscale image I
Output: threshold value τ

h ← COMPUTEHISTOGRAM(I)
m0[0] ← h[ℓ]
m1[0] ← ℓ * h[ℓ]
for ℓ ← 1 to ζ − 1 do
    m0[ℓ] ← m0[ℓ − 1] + h[ℓ]
    m1[ℓ] ← m1[ℓ − 1] + ℓ * h[ℓ]
μ ← m1[ζ − 1]/m0[ζ − 1]
σ̂b² ← 0
for ℓ ← 0 to ζ − 1 do
    σb² ← (m1[ℓ] − μm0[ℓ])²/(m0[ℓ] * (m0[ζ − 1] − m0[ℓ]))
    if σb² > σ̂b² then
        σ̂b² ← σb²
        τ ← ℓ
return τ
```

Image Segmentation

- **Adaptive Thresholding**

- Global thresholding techniques do not perform well when the image noise characteristics vary across the image.

- To overcome such difficulties, adaptive thresholding techniques are needed, in which the threshold used at any given pixel in the image is based upon ***local statistical properties*** in the neighborhood of the pixel:

$$I'(x,y) = \begin{cases} \text{ON} & \text{if } I(x,y) > \tau(x,y) \\ \text{OFF} & \text{otherwise} \end{cases}$$

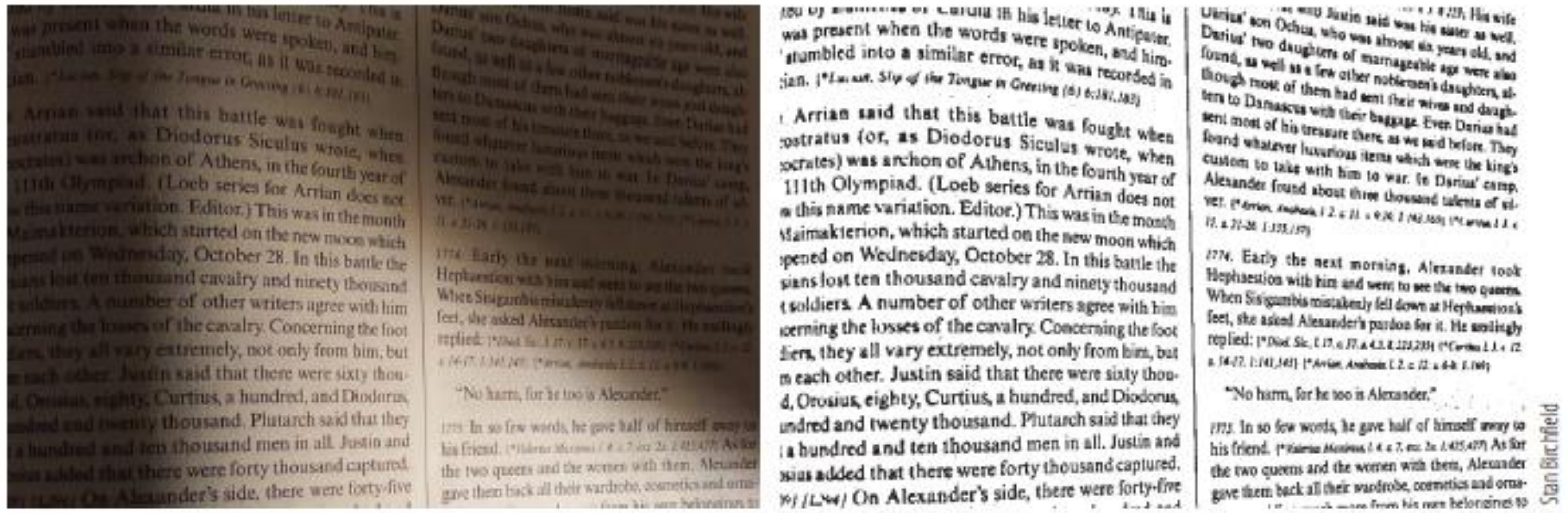


Figure 10.4 Example of adaptive thresholding.

Image Segmentation

- **Adaptive Thresholding**

- **Chow-Kaneko method:**

  - The image is divided into overlapping blocks, and the histogram is examined using global thresholding methods for each block to determine a threshold value for the block.
  - Interpolation between these threshold values then yields a varying threshold function defined over the entire image.

- **Niblack's method:** based on the block mean μ and standard deviation σ

$$\tau(x, y) = \mu(x, y) - k \cdot \sigma(x, y)$$

Image Segmentation

The plethora of segmentation techniques have been proposed:

- Deformable Models (based on Energy Minimization):
  - Active Contours (Snakes)
  - Gradient Vector Flow
  - Geodesic Active Contours

- Segmentation into multiple regions (e.g. semantic segmentation):
  - Splitting & Merging
  - Region Growing
  - Watershed Method
  - Mean-Shift
  - Graph-based methods (Normalized cuts, Markov Random Field, Conditional Random Field, etc.)

- *These techniques have been outperformed by Deep Learning!*

Model Fitting

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- We want to find a model which describes the observed data points.
- The data points can lie simply in **2D image space (x,y coordinates), in 3D space (x,y,z coordinates)** or in the 'feature space' (RGB color, grayscale values, region eccentricity, region compactness, SIFT, HOG, etc.).
- 'Fitting a model' means to find the model which would describe the observed data points (2D/3D coordinates, Features) the best.
- *Have we done any model fitting so far? Or have we fit any (data) models so far?*
- *Any parallels with Machine Learning? Think about classification and feature space.*

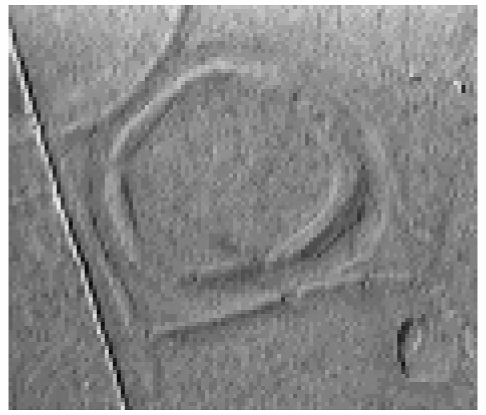
Model Fitting

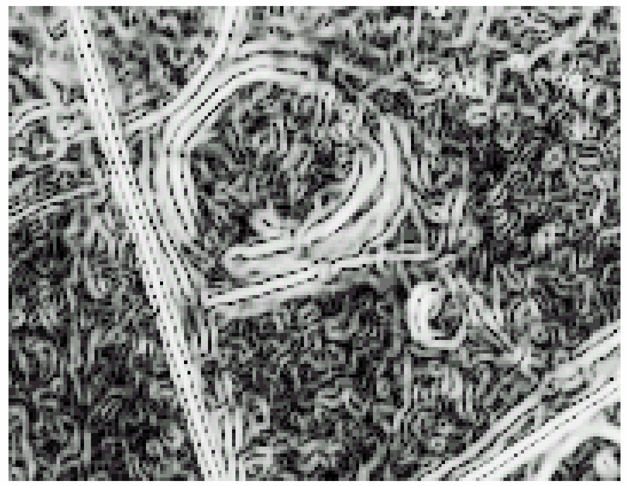
- *How do you find geometric shapes (*corners*, lines, curves, ellipses, circles) in an image (sometimes also referred to as 'template matching'?*

- Template matching leverages **higher-level domain knowledge in the form of** the model or template (line, plane, circle, curve) we want to fit

- Template matching exhibits robustness to noise due to this prior domain knowledge about the shape we are looking for

- E.g. automatic segmentation of sub-circular objects (ringforts)
- Laplacian-of-Gaussian
- Manual thresholding of LoG image
- Canny (lower=50,upper=200,sobelsize=3)





A motivating example for template matching

Model Fitting – Hough Transform

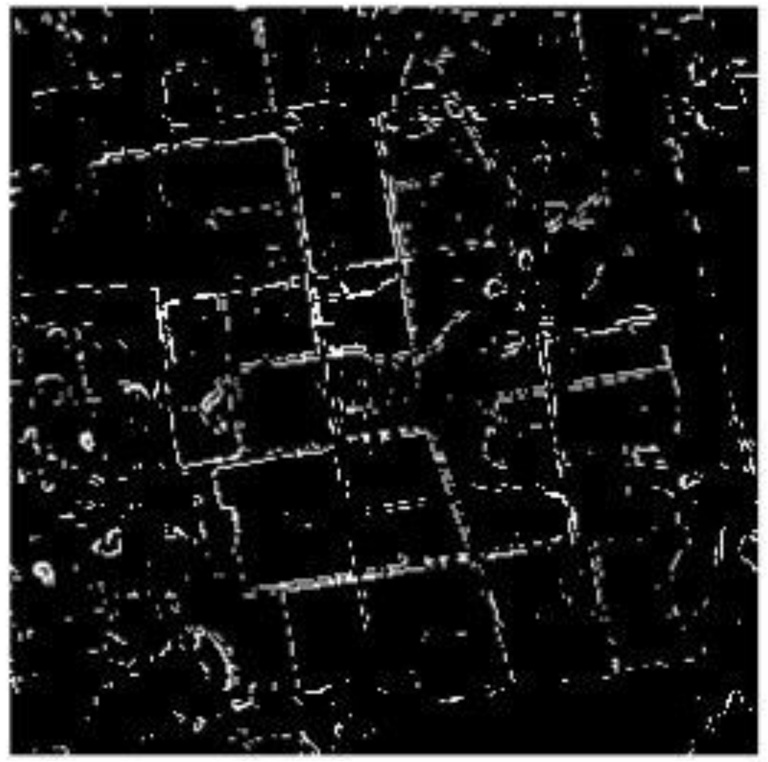
- What are Transforms? represent stuff in a “new” parameter space

- Hough Transform represents 2D image points in the spatial domain to a parameter space, with each dimension representing 1 *parameter of the shape*

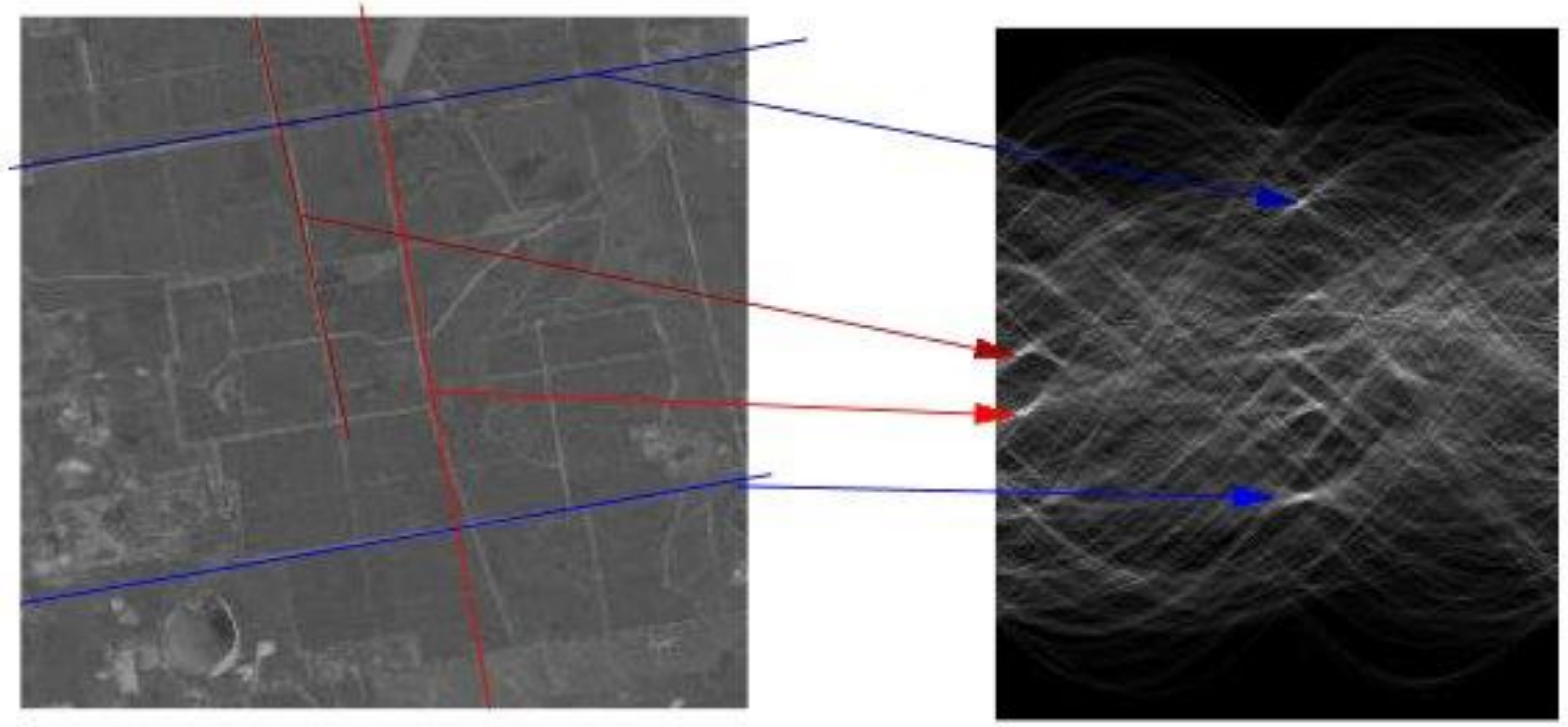
1. **Find a suitable parameter space**
2. **Calculate Hough Transform for the given image**

3. Search for peaks in the transform/parameter space

4. The shape corresponding to the peak parameters in the parameter space gives us the best fitting model/template (line, circle, etc.)



Thresholding Sobel filter responses yields broken lines



Hough transform with θ on horizontal axis and *radius* ρ on vertical axis

Model Fitting – Hough Transform

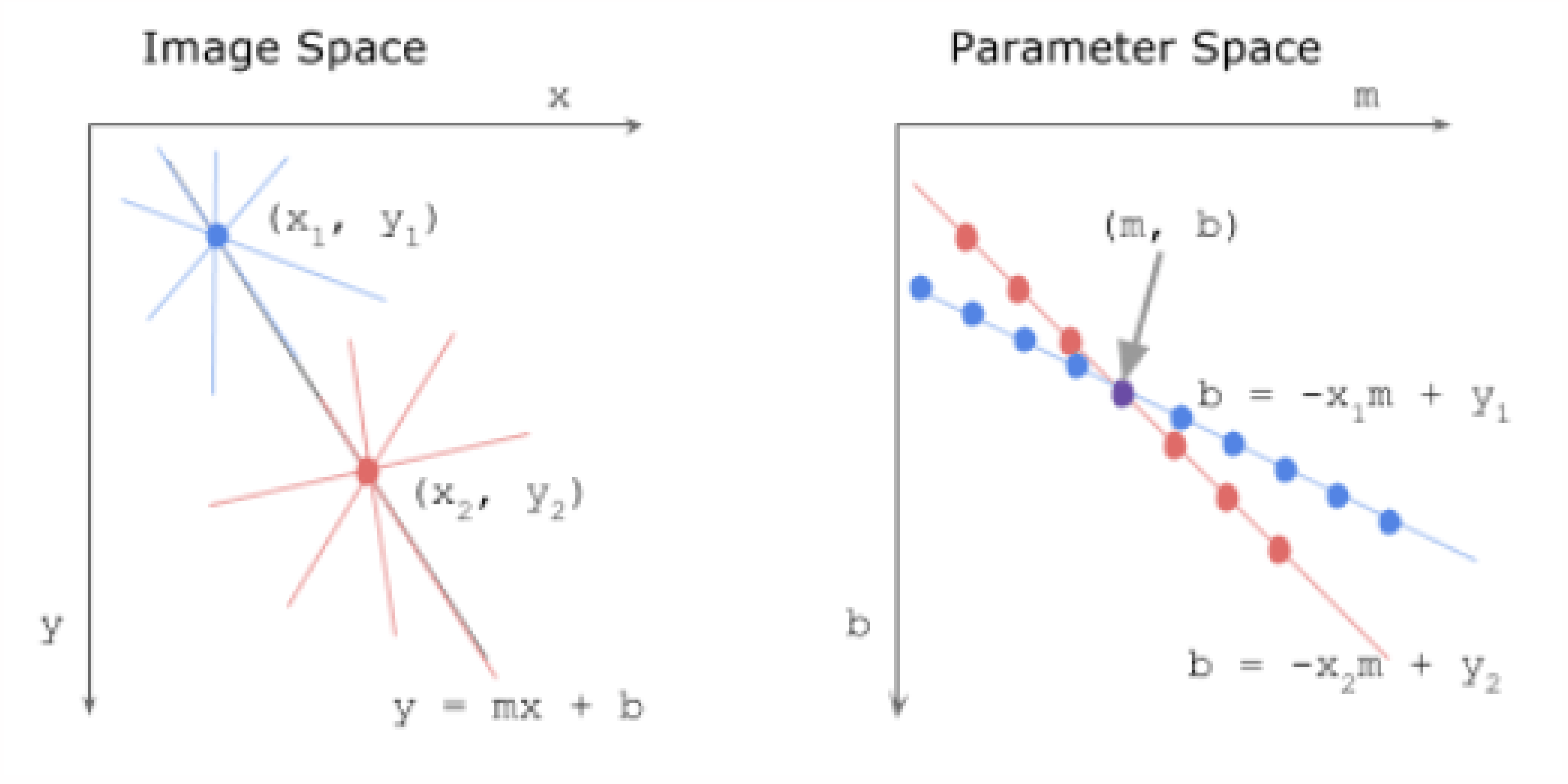
Ollscoil na Gaillimhe
University of Galway

Hough Transform for Lines: Finding a parameter space

- toughest step to conceptualise!

$$y = \underbrace{m}_{\text{slope}} x + \underbrace{k}_{y-\text{intercept}}$$

$$b = \underbrace{-x}_{\text{slope}} m + \underbrace{y}_{y-\text{intercept}}$$

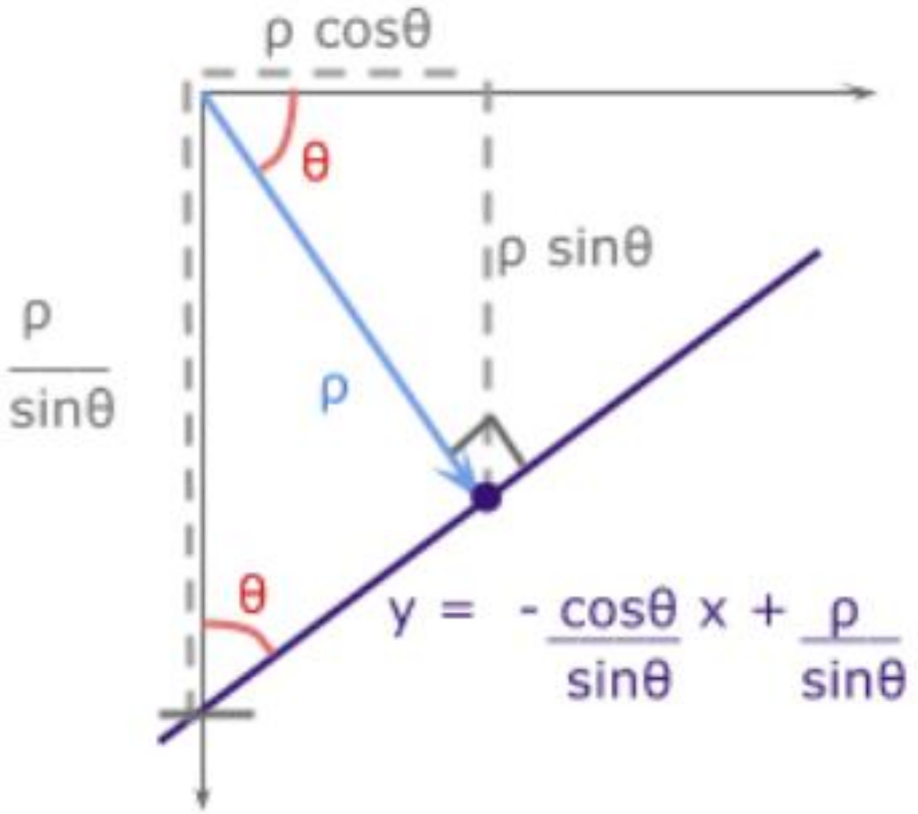


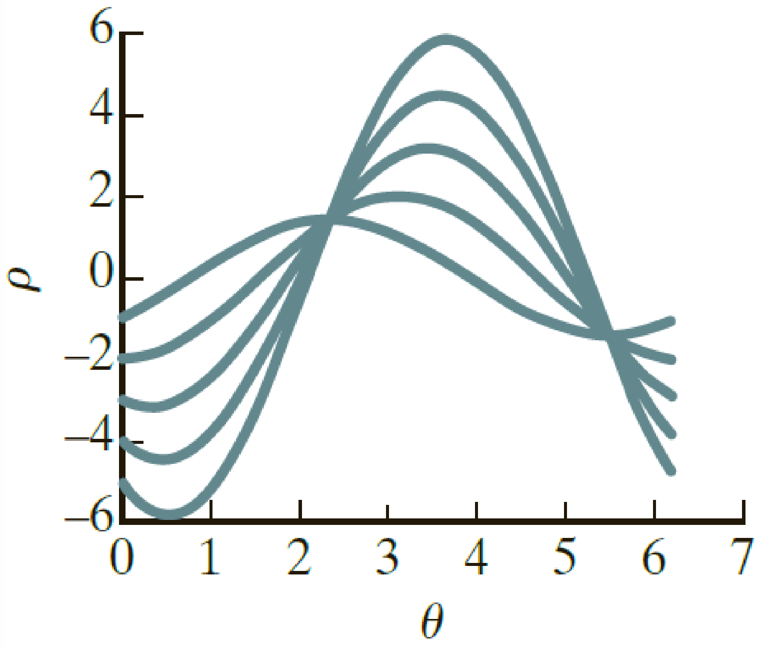
- *What is the problem with the above parameter space?*

- The new parameter space for lines (written in two different ways at times)

$$\rho = xcos(\theta) + ysin(\theta)$$

$$x \cos\theta + y \sin\theta + \rho = 0$$



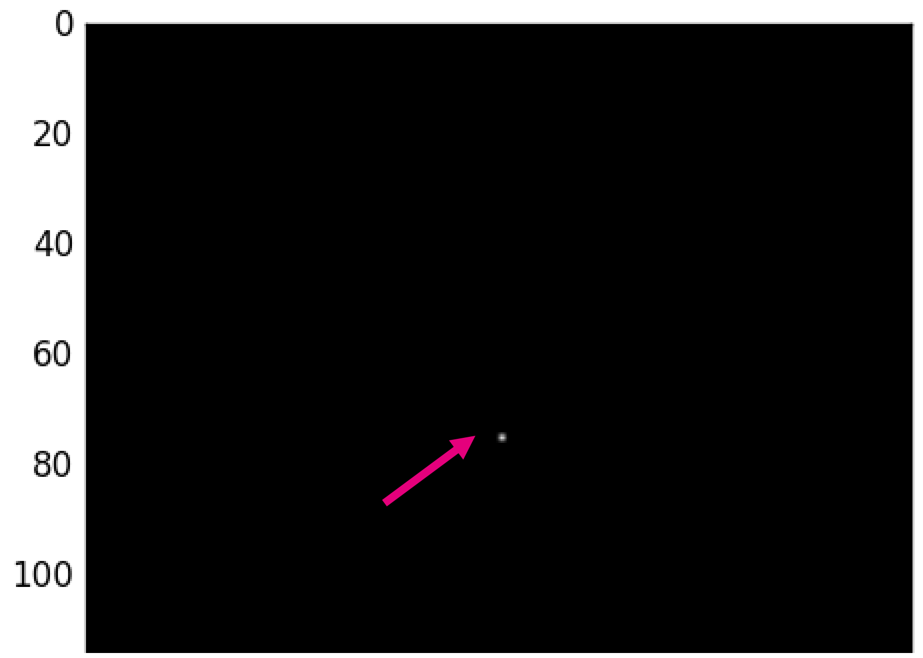


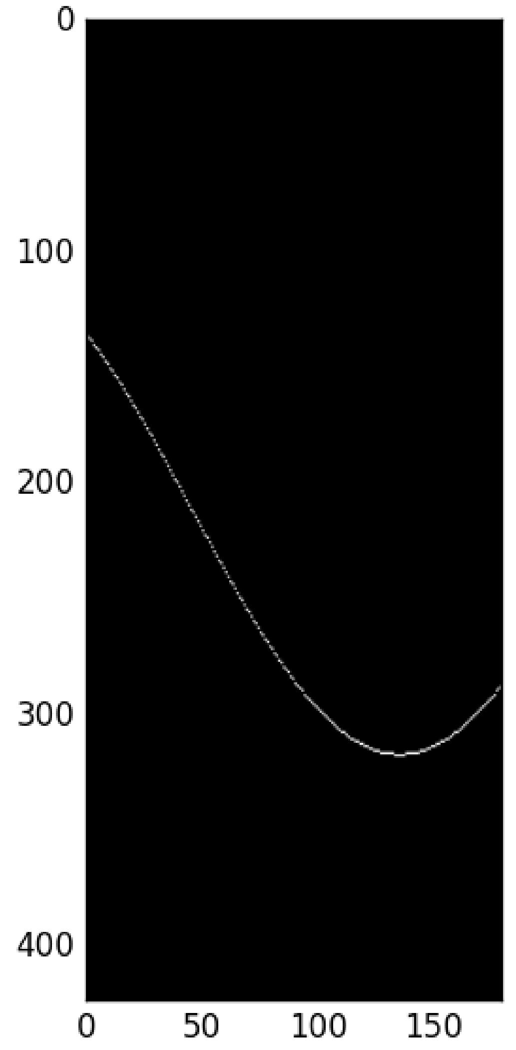
Source: https://sbme-tutorials.github.io/2021/cv/notes/4_week4.html

Model Fitting – Hough Transform

Hough Transform for Lines: Calculate the Hough Transform/Fill the parameter space

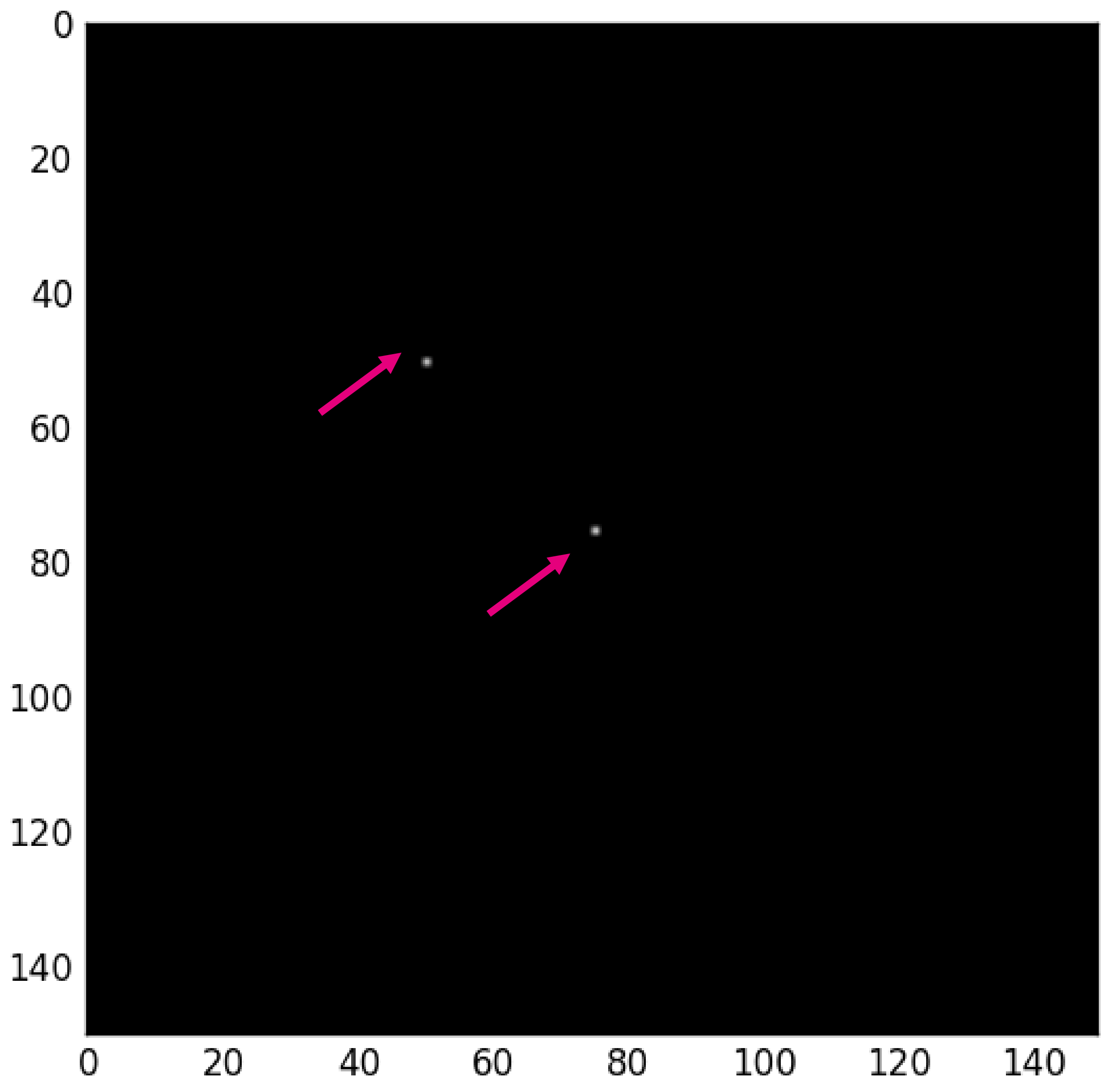
- Each **edge** pixel casts a single 'vote' for each of the lines that it may belong to
- 'Votes' are gathered in an accumulator array, with one array dimension required for each parameter of the shape being sought
- Peaks in the accumulator array are sought, and the lines with the most 'votes' are thereby identified and extracted
- Challenge is the selection of discretization of the parameter grid and selection of the threshold for the parameter space

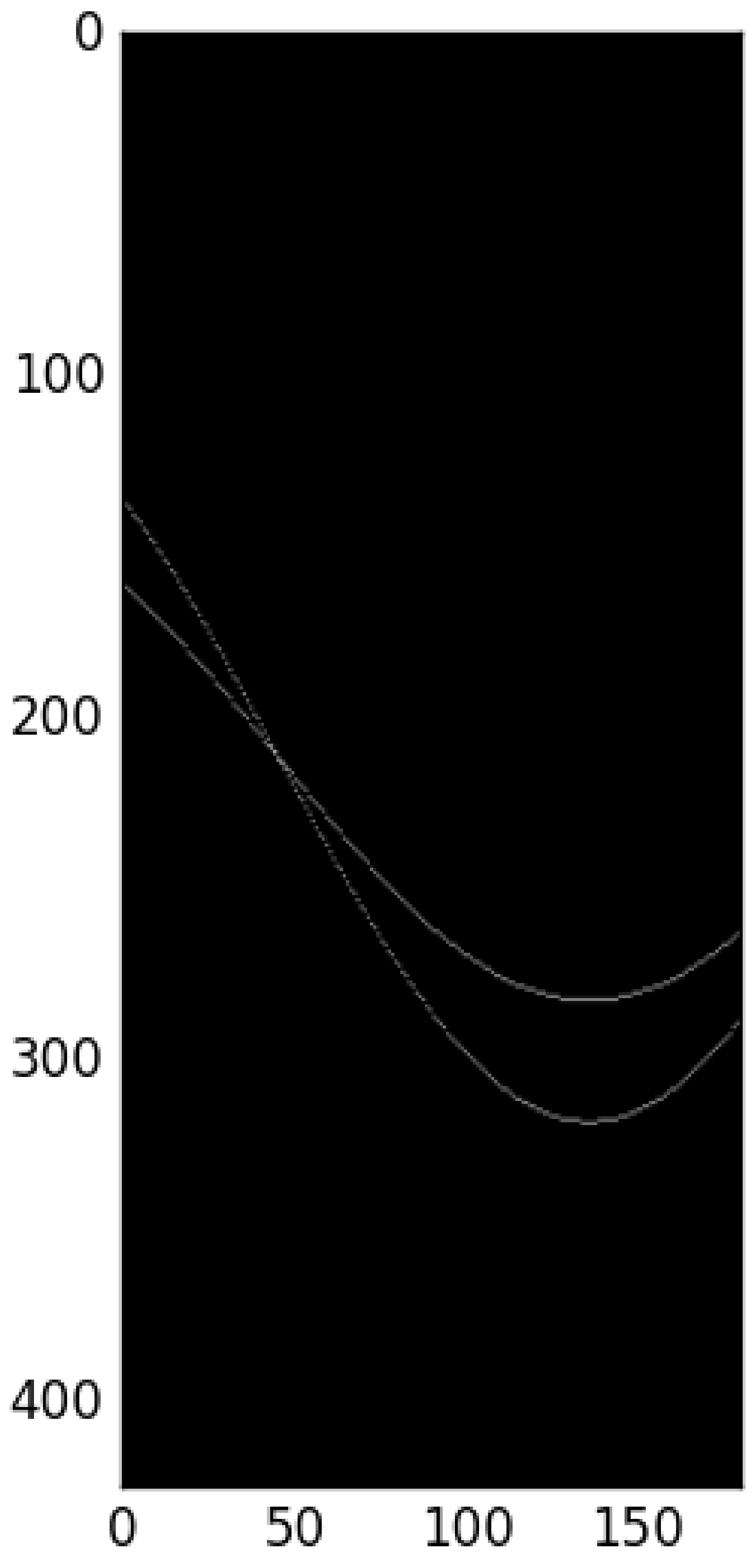




Source: https://sbme-tutorials.github.io/2021/cv/notes/4_week4.html

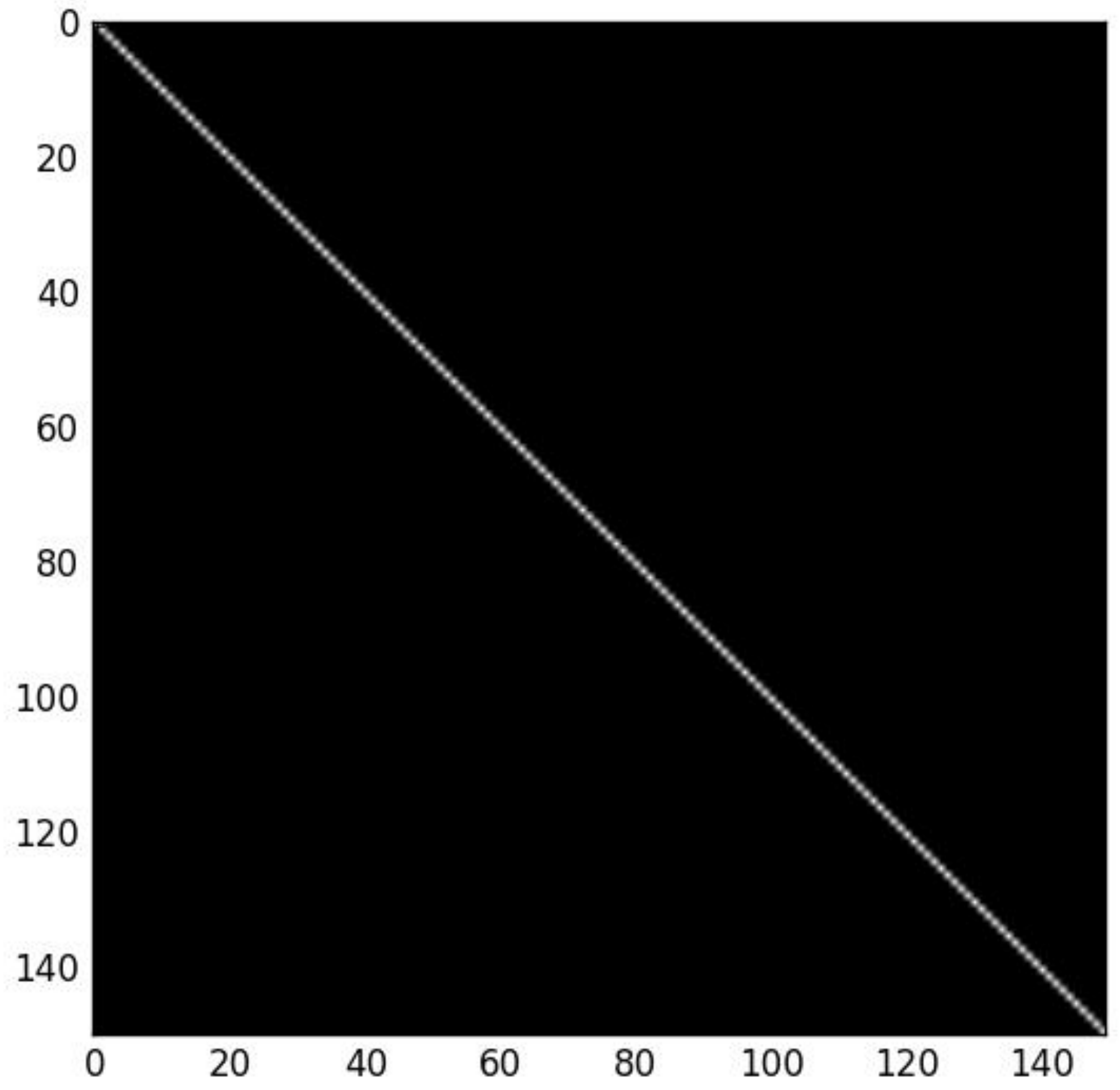
Model Fitting – Hough Transform

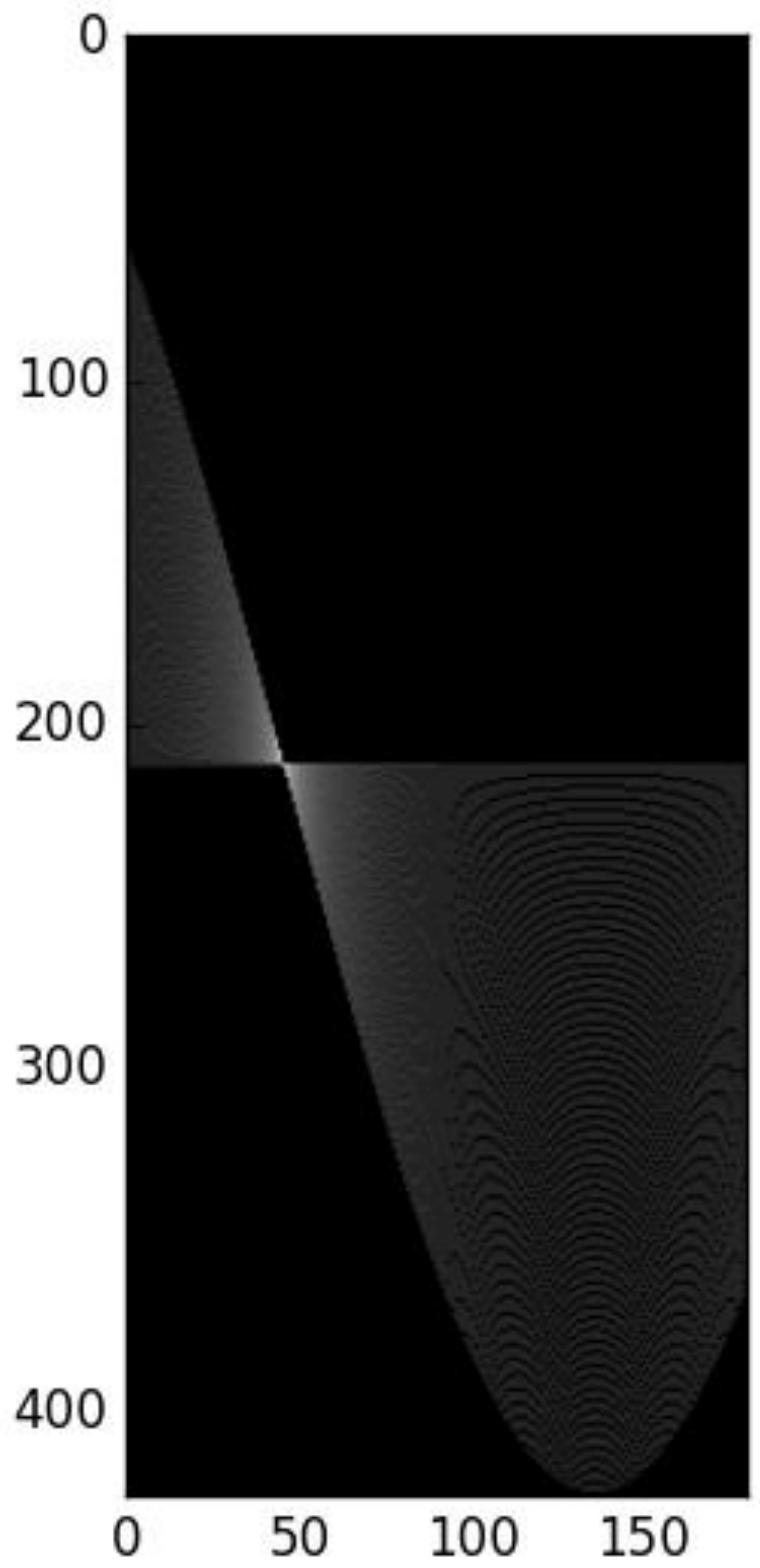




Source: https://sbme-tutorials.github.io/2021/cv/notes/4_week4.html

Model Fitting – Hough Transform

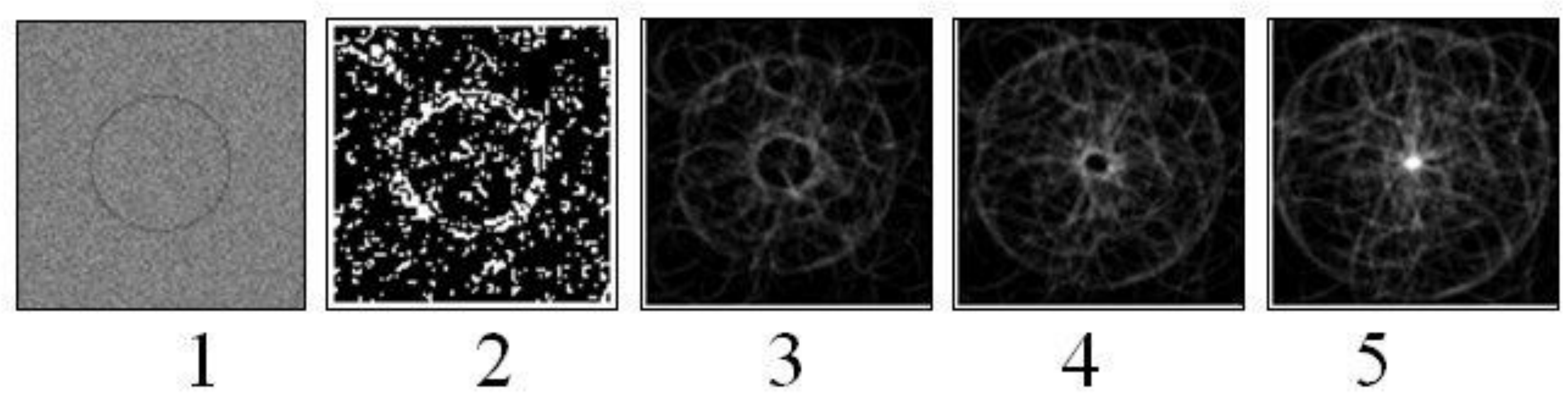




Source: https://sbme-tutorials.github.io/2021/cv/notes/4_week4.html

Model Fitting – Hough Transform

- **Hough Transform for Circles:**
- A **3-**dimensional parameter space: x location of centre, y location of centre, and radius
- Example: a circle HT of a noisy image containing a low-contrast circle of radius 24 pixels:



(1) the original image, (2) the edge map used by the HT (following smoothing, edge detection, thresholding), (3) accumulator array at radius 15, (4) accumulator array at radius 20, (5) Despite the presence of noise in the object's detected boundary, the accumulator array shows a strong peak at the centre of the circle at radius 24.

Model Fitting – Point Cloud Matching

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

- When we want to fit/match/register/deform one point cloud to another 2D/3D point cloud
- Applications range from lidar point clouds in robotics to 3D graphics

- Suppose we have two sets of 3D points:

$$\{(x_i, y_i, z_i)\}_{i=1}^{n} \qquad \{(x_i', y_i', z_i')\}_{i=1}^{n} \qquad \mathbf{x}_i = (x_i, y_i, z_i) \qquad \mathbf{x}_i' = (x_i', y_i', z_i')$$

- **Procrustes analysis:** Find the transformation so that when we **stretch, shift, and rotate** one point set, the result matches as closely as possible to the other point set, i.e. to find *s*, **t**, and **R** that minimizes the following sum of squared errors (a least-squares problem):

$$\min_{s,\mathbf{t},\mathbf{R}} \sum_{i=1}^{n} \|(s\mathbf{R}\mathbf{x}_i + \mathbf{t}) - \mathbf{x}_i'\|^2 \qquad (11.78)$$

- **Iterative Closest Point (ICP)** algorithm: An iterative Procrustes analysis to solve the more difficult problem when the correspondences between the two point clouds are unknown or the number of points in the two point clouds are different.

Next Time

- 3D Computer Vision starts with camera calibration
- With lecturer Dr. Waqar Qureshi

Ollscoil na Gaillimhe
University of Galway

Thank *you*